IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	Davis
Serial Number:	10/823,871
Filing Date:	April 14, 2004
Art Unit:	2145
Examiner:	Mirza, Adnan M.
For:	**Method of load balancing edge-enabled applications in a content delivery network (CDN)**

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RESPONSE TO NON-FINAL OFFICE ACTION

This paper is submitted in response to the office action mailed May 2, 2007. A three (3) month extension of time is submitted herewith to extend the response deadline through November 2, 2007.

REMARKS

Claims 1-18 were rejected under 35 U.S.C. §102(e) as being anticipated by Melchione et al, U.S. Publication No. 2004/0006586. Respectfully, this rejection is traversed.

Melchione et al describe a system for distributing software components to nodes distributed throughout a network. As illustrated in Figure 2, an application service provider provides services for administrating instances of the software 212 via a data center 232. A particular instance of the software might be a software release that needs to be delivered to one or more of the nodes in the network. Each node includes a computer 224 having an agent 228 that communicates with the data center 232 to assist in the process. According to paragraph [0070], the distribution of the software 212 can be accomplished in several ways. In one approach, an administrator specifies which nodes are to receive which releases. Automated processing can

then be used to distribute the software to those nodes, and the source copy of the software (that is to be distributed) may reside anywhere in the network. Thus, if desired, a node can be designated to provide the software to another "peer" node within the network, thus alleviating the burden of distributing the software releases from a central location (which could end up as a bottleneck). This particular operation is described in paragraph [0135], which indicates that a "central database can be employed to enable agents to obtain software from other agents."

The subject invention is not a software distribution mechanism. Rather, at a high level, the present invention is concerned with a completely different problem, namely, how to load balance application server resources operated in a distributed set of servers (e.g., the distributed servers of a content delivery network). As described (see, e.g., the preamble of claim 1), each server in the set typically includes a server manager process, and an application server on which applications or application components are executed. As service requests are directed to servers in the set, the application servers manage the requests in a load-balanced manner, and without any requirement that a particular application server be spawned on-demand.

How this is accomplished is described by the operating environment and functionality as shown in Figure 4. As seen there, each server 402a-402n in a given server region 400 includes several processes: a manager process 404, a monitor process 406, and an aggregator process 408. The aggregator process 408 keeps state as to which web applications are running on which server, and it publishes this information to each server manager running in the region 400. In the context of representative claim 1, this information is the "first data set identifying which application components are actually loaded." In other words, the "first data set" tells each server manager "what is loaded where" in the region. The aggregator process 408 also implements a load balancing algorithm from which the output is a "second data set" (see representative claims 3-18) that tells the server manager "what [applications or application components] should be loaded where" in the region. This and it allows the CDN server manager to implement an application loading policy based on this second data set. In particular, using the second data set, the aggregator process 408 can ask a given server manager running on a given machine to load a application, e.g., by adding that application to the second data set for that server manager, or by "mapping" that application to that server manager.

In contrast to Melchione et al, which deals with distributing software to nodes, the subject

disclosure is primarily addressed to re-directing service requests from one node (at which the request is initially received) to another node (at which the desired application component is available and running). This "re-directing" of service requests is seen in each of independent claims 1 and 7, and in dependent claim 14, as evidenced below:

claim 1

"if the instance of the given application component is not loaded on the given server, directing the service request to another server in the set of servers as a function of information in the first data set"

claim 7

"if the instance of the given application component is not loaded on the given server, directing the service request to another server in the set of servers in the given server region as a function of information in the first and second data sets"

claim 14

"wherein the code for balancing load implements a first policy with respect to service requests that cannot be serviced by a given manager process at a given server" (from claim 13)

"wherein the first policy directs a service request from the given server to another server that, as indicated by information in the first data set, the given application component is loaded."

Melchione et al teach that an agent on one node may access a "central database" and thereby learn about peer nodes that can provide a particular software release. This is not what Applicants are claiming here. Rather, as indicated by the underlined portions of these representative claims, the subject disclosure is directed to directing a service request from one node (where it cannot be serviced) to another node (where presumably it can be). In addition, several of the subject claims (e.g., claim 12) address the further goal of "balancing load across the set of servers" based on information in the first data set ("what is loaded where") and the second data set ("what should be loaded where"). There is no such concept remotely disclosed or suggested in the cited reference.

Melchione et al are distributing software to nodes that do not have that software; as noted above, most of the subject claims deal with redirecting client requests for service from nodes that cannot service those requests to nodes that can, preferably in a manner that (according to claim 12) balances application server loads across the set. Melchione et al are not teaching load

balancing at all; at most they describe using additional resources (e.g., peer nodes) to help in distributing the software. While these nodes can alleviate the bottleneck associated with a central server-based distribution, the actual load balancing aspects of the relevant claims are not seen in the reference.

The Manual of Patent Examining Procedure (MPEP) § 2131 provides that a "claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described in a single prior art reference." While it is true the literal correspondence of the wording need not be shown, the "identical invention must be shown in as complete detail as contained in the ... claim." Melchione et al does not meet this rigid requirement because the reference does not disclose at least the "re-directing service requests (claims 1-11) or "load balancing (claims 12-18) subject matter referenced above, let alone the use of the "first data set" (alone or in combination with the "second data set") to accomplish these functions.

Respectfully, the Examiner is asked to reconsider and withdraw the rejection, as there is no anticipation of any pending claim.

For the above reasons, reconsideration and favorable action are respectfully requested.

Respectfully submitted,

By: _____

David H. Judson, Reg. No. 30,467

ATTORNEYS FOR APPLICANT